LOADSTAR LETTER #48

UPS Strike Hits CMD

By Jeff Jones. In recent talks with CMD it was revealed that the UPS strike caused a lull in phone orders at CMD because customers mistakenly believed that CMD would continue to ship primarily through UPS during the strike. CMD wants everyone to know that if this strike is still on while you're reading this, they're using the US mail and other means for now.

Subterfuge and Shady Dealings: The Collapse of Commodore

By Jason Compton. Commodore, as we knew it, was a successful pioneer of home computers. Few companies can claim that they have done as much to innovate and proletarianize the idea of affordable, powerful computers. In the 1970s and 1980s, Commodore again and again set the standards.

Unfortunately, their run came to an end in April 1994, when uncontrollable losses and a maxed-out line of credit put the company into bankruptcy proceedings. Since then, the Commodore intellectual assets have traded hands more than once, and a whole new world of interesting back-room stories and anecdotes have come to light. Some are mere speculation based on available information. Others are matters of public record. It's always a bit unfair to air someone's dirty laundry when they can't defend themselves, but in this case, the fun of the read outweighs the possible moral implications. :)

The Commodore bankruptcy lasted almost an entire year, April 1994 to April 1995. Escom of Germany, a big PC cloner in Europe, acquired virtually all of Commodore's physical and intellectual properties, excluding the various buildings and offices which had previously been sold. Not surprisingly, a lot of Amiga goods were part of the sale, but what's perhaps less well known is just how much 8-bit equipment went into the deal. Exact numbers are not available, but suffice it to say that companies like Paxtron, Software Hut, and CMD got considerable influxes of raw and complete material as a direct or indirect result of the sale of Commodore assets. The Commodore US office auction flyer showed a room with 1571 drives stacked 10 high and several deep on a desk!

The auction process took an unexpected detour when it was revealed that the liquidator of Commodore's German subsidiary had made a deal with Escom to sell them the Commodore trademark. When (and only when) the Commodore International liquidator objected to the legality of such a deal, Escom made a bid for the entire Commodore holdings.

The liquidator of Commodore International, Franklyn Wilson, was an officer with Deloitte and Touche, the accounting firm that was assigned to Commodore's estate. But Wilson, in a deal still not fully understood, left D&T midway through the proceedings and took the Commodore account with him to start his own private firm.

The liquidator of Commodore Germany, Bernard Hembach, who made the challenged deal for the Commodore trademark with Escom, went on to be Escom's liquidator when they themselves went bankrupt in July 1996.

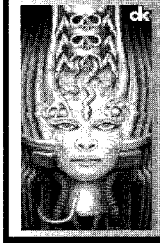
Commodore Chairman Irving Gould is a very interesting man. In 1983, he was #9 on a list of the world's wealthiest executives. Despite having a long history and a significant holding in Commodore, his real money comes from international shipping, his Pacific Trade Associates firm. It is said that to avoid any sort of income tax on his vast earnings, he was made a citizen of an African nation after doing a favor for the government. The country in turn does not tax his income, and further he spends no more than 72 hours at a time in any one country, to avoid any taxes they might be able to impose. He apparently lives on a private jet. (I have good reason to believe he may have been one of the inspirations for the H. R. Hadden character in Contact.)

Irving Gould wasn't just Commodore's chairman and plurality stockholder. He was also a significant creditor through various loans he and Pacific Trade Associates had made to the company. It is rumored that in lieu of waiting for cash through a bankruptcy settlement, Gould accepted two 18-wheeler loads of Commodore CD32 game consoles as a loan payment. He is alleged to have turned around and sold these units to a wholesaler in Germany for $90 apiece. There is no direct proof of this transaction, but around the same time the retail price of a CD32 dropped from $400 to about $230.

Not all of Commodore's intellectual properties were sold to Escom. A handful of patents (including a rather bizarre patent on a device to control watertight doors held by Commodore France) and some game trademarks and code rights, largely for the VIC-20, were not a part of the original sale. But the question of where exactly these obscure rights went, and why they were not offered to Escom, remains unanswered.

Shortly before Commodore's bankruptcy, the company took out large misconduct insurance policies on its top officers, including CEO Medhi Ali. This did not go over very well with

Contents Of This Issue



 The LOADSTAR LETTER is published monthly by J&F Publishing. 606 Common Street, Shreveport LA 71101. Subscription rate is $18.00 12 issues. Contacts:

jeff@LOADSTAR.com

wookie@inconnect.com

US MAIL: ATTN. Jeff Jones
J & F Publishing P.O. Box 30008, Shreveport, LA 71130-0008, Phone: 318/221-8718, Fax: 318/221-8870, BBS: 318/425-4382

Commodore's creditors, but in the end it held up and the creditors were forced to make due with only a $12 million payoff from Escom on a consolidated debt of about $140 million.

Mr. Ali has been spotted at various computer conferences around the world since 1994, usually promoting PC clone products.

General Alexander Haig, whom many of us remember for his "I am in charge" speech in the early 80s, was a Commodore board member. This has been grist for many a mill, generating unconfirmed reports that Haig used Commodore for gun running. Haig has been spotted since in Forbes magazine, heading up a venture to create cellular networks in cities based not on ground-based transmitters but on dirigible-mounted antennae.

After Escom's bankruptcy, their Dutch branch managed to stay afloat and raise enough money to start up their own independent company. They also got enough money to buy the Commodore trademark from Escom, meaning that today you can walk into a Commodore store and buy a Commodore Pentium machine.

Commodore BV, the name of this new cloner, is itself being purchased by Tulip Computers. Tulip believes that this move will make the joint business larger than the sum of the parts, and the new company will be among the 10 largest computer firms in Europe.

Bernard Van Tienen was the man who spearheaded Escom's efforts to acquire the Commodore properties from the liquidator. He was then placed in charge of Escom's Dutch branch, and spearheaded their efforts to buy the Commodore trademark from Escom. In effect, Mr. van Tienen has the unusual distinction of being the man who has bought the same trademark twice in less than 24 months.

Jason Compton, jcompton@xnet com Editor-in-Chief, Amiga Report Magazine
Anchor, Amiga Legacy http://www cucug org/ar/
http://www.xnet com/~jcompton/

Rumors Of A New Super Snapshot?

By Jeff Jones. Joe Palumbo (JP PBM) has purchased the rights to Super Snapshot, and is currently taking orders for the cartridge. I've searched the Internet for more information and it is scant though he reportedly purchased the rights months ago. Doug Cotton was quoted on Usenet attesting to the following of Super Snapshot: "Orders are already being taken, and it looks like shipping is supposed to begin around the beginning of September. So far as I know, Joe isn't on the net, or I'm sure he'd probably have dropped in here to comment on it himself. And while 5.2 may have been the last version LMS ever made, it isn't a certainty that it will be the last version ever since the rights have changed hands. When he obtained the rights a couple of months ago, Joe contacted us for suggestions to modify and update the cartridge. Since he works closely with (and is a member of) TPUG, and since Doug Roger of TPUG had been in some way involved with work on a possible version 6 while LMS still held the rights, I suspect that Joe and Doug may now be working together on bringing some of that about. Doug did appear to know the product very well in conversations I had with him at various Commodore shows in Canada, and the last time I was up there Doug and Joe were together when I ran into them."

New Centipede BBS Software

By Gaelyne Gasson. This info was sent to me from a BBS sysop that uses the CommNet Network, and I felt it should be passed on. I've known about Centipede BBS now for about 3 years and have been patiently waiting for it to be released. It's nice to see it's finally reached this stage.

Centipede is a revolutionary new BBS software package for Commodore computer systems. Taking advantage of the latest in computer technologies from the Internet and over 50 years of combined programming and BBS experience BUGSOFT has unleashed the Centipede 128 BBS. Features include:

- Hardware Requirements
- Commodore 128 or Commodore 128D Computer
- 80 Column Monitor
- 1200 to 33.6 Modem (Hayes Compatible)
- Disk Drives or Hard Drive (RAM Unit or HD required)
- Modem Interface SwiftLink/Turbo 232 Supported Phone Line
- Pricing: CENTIPEDE BBS Software - $69.95 + $9.95 S&H
- V128 Upgrades to Centipede - $59.95 + $9.95 S&H
- Runs on a Commodore 128 in C128
- Supports LTK
- Supports from 1200 to 230
- Graphics - 80 column C/G
- Supports up to 15 Separate Message boards (local
- Each message board supports 26 threaded or unthreaded Categories.
- Supports up to 15 U/D Categories with 26 directories each
- 300 files per directory to 254 directories.
- Simulated 40 columns for C64 users
- Scripting
- Polymorphic Menu System
- Unlimited Online Gaming Abilities. (More to be added)
- Protocols Xmodem
- Easy to install add-ons
- ComLink Nationwide Messaging Network (CommNet coming soon)
- Net64 (Color 64 Network)
- Online Help Files
- Extensive message board editing and control.
- E-mail quoting and file attachments
- Superior SysOp Control features that are built in and EASY to use.
- Local Buffer
- Built in Term program
- Multiplexing on a LTK dual line system.
- Cross-line chat capable with Multiplexed system
- Individualized User Access Control.
- Split Screen Chat Mode
- Automatic Detection of peripherals (CMD HD RAMLink LTK.)
- Full Screen Text Editor
- Detailed Caller Log
- Very fast system all the way around.
- Full Screen U/D File Selector
- Very programmer friendly easy to write your own modules
- Programmable Alt Keys
- Nine access levels with individual adjustments available.
- Automatic accounts purging
- User account application on call
- Up to 1000 lines per message
- Local Mode Use password
- Error recovery Module
- Upcoming Features
- CommNet Network
- Internet connectivity (Email and more)

Some technical notes from the programmer can be found at http://www.bugsoftware.com.

Owner Support is FREE via Email on the Internet ComLink direct at
Nature Reserve
Adam Fanello of BugSoft
4822 Larwin Ave.
Cypress CA. 90630-3515
Nature Reserve BBS (714)-828-7296 and (714)-952-2696 (Handle ANT)

Maurice Randall Pleads Case For New Hardware

By Maurice Randall on Fidonet. Maurice's essay is in response to the following excerpt from a Commodore Purist on the usefulness of programs written for CMD's new RAM cards:

*"...However, such a program would not run on a *real* C64. So using this card, you turn your computer into a completely new machine that does not remain a C64. When this card is accepted in the C64 scene, we get into a situation the C64 has been preserved from for a long time, a situation you can find on all the 'big' computers that are off topic in this area: If you want to run the latest software, buy the latest hardware..."*

When Commodore designed the 64, they designed it so that after-market companies could create new things that could plug into the machine. I don't understand why some people think that these ports should remain empty! It makes me think that some people would rather prefer that Commodore never included a user port or a cartridge port. If the 64 were never made to be expandable, Commodore would have really flopped on this one.

New hardware normally follows with new software. Why is that so hard to accept? If you don't have one, you do not need the other. But if you have one, such as the hardware, you just might want the software that is created for it. And believe it or not, a 64 with a SuperCPU plugged in is still a 'real' 64. Have you ever tried to power up a SuperCPU without first plugging it into a 'real' 64? (Or a 128). You certainly can't use a SuperCPU on some other computer that is running a 64 emulator. Stop and think about that software issue for a moment. Try running geoLaser and geoPubLaser without a PostScript printer. (I'm not talking about the patched version for making a disk file). Isn't that the same thing? Does having a PostScript printer connected to a 64 make the 64 no longer real? After all, the software is useless without the PostScript printer. As far as I am concerned, it is a real 64 doing a real computing job. GeoFAX requires a SwiftLink or Turbo232. Does this kill the 64's 'real' status?

That reminds me of a previous argument that requires an old Commodore modem to be plugged into the user port in order to keep the 64 in its 'real' status. Isn't that still some form of hardware plugged into the machine? What difference does it make how much hardware is plugged in and how extensive that hardware is? My 64 is still a 'real' 64 even though I have a SwiftLink with a 28.8 modem, a CMD-HD, an FD-4000, a geoCable connected to a PostScript printer, and a SuperCPU complete with its new RAMCard. None of this stuff will work with any other computer, except for the drives. All this arguing over the 64 is pretty silly. I say, run the thing however you want. That is how Commodore designed it. It is a very flexible and expandable computer. One of the best ever made.

Multitasking On A C-64

By Ian Moote from an excellent post on Fidonet with a foreword from Jeff Jones. There's a big debate on Fidonet about multitasking. Many people say it can't be done on a C-64. Definitely not true. The bulk of the debate seems to be the *definition* of multitasking. Ask ten people what multitasking is and you'll get twelve answers. When you get into complicated subjects such as this, lines get blurred and what's true to one authority may be shot down by another. As far as I'm concerned, scratching your neck while you walk down the street is multitasking. By that same token, making the cursor flash beneath the ready prompt while scanning the keyboard and updating the jiffy clock is multitasking in my book, whether or not the tasks are run on separate IRQs.

Depending on what you call a "task," any computer and most any operating system multitasks because many important tasks are being taken on at the same time "in the background." In other words, your computer is nearly always doing more than you think it's doing. True, a single CPU is doing only one thing at one time, but every sixtieth of a second, your C-64 takes off and does "housekeeping." Well that housekeeping is a collection of sophisticated subroutines. Your C-64 checks the keyboard for key presses, updates the screen, updates the clock and many other things even when a program doesn't ask it to.

Consider that Mr. Mouse is a program that senses mouse movement, represents it graphically on a screen according to presets, and even updates variables for you. *That's* a task. Moving that sprite all over the screen means that while another program is running, Mr. Mouse is indeed also running. If a program is busy crunching numbers while the mouse is freely zipping around the screen, many things are being done at once. This isn't considered true formal multitasking by some because they say it's only a routine being called every jiffy off of the IRQ. Mr. Mouse isn't a *named* program that's a list of multiple tasks taken on by an official multitasking OS like Windows 95.

People tend to think that multitasking is a function of a computer. The Amiga was known as the multitasking giant of the 80s. The Amiga had separate processors for graphics and sound, which allowed tasks to be accomplished cleanly and quickly. With little or no help from the main CPU, a stock Amiga could blast one million pixels on the screen in record time. This isn't much different than the separate processors in your very smart 1541/71/81/CMD drives or a decent graphics/sound card in a PC. These separate processors all work together to get the job(s) done faster. None of this made the Amiga any better at multitasking than a C-64. The Amiga's limited multitasking is a function of its operating system. What follows is an excellent dissertation on multitasking by Ian Moote. It is preceded by the comment that prompted him to post.

Multitasking: Performing more than one task simultaneously. Time-sharing: Performing more than one task sequentially (albeit automatically, i.e., DOS wedge, background music, etc.) Is this close to being correct? And if the multitasking definition IS correct, would it not require multiple processors, or at least multiple program counters and other registers that Commodore 64 128s just don't have?

I don't normally get into this discussion anymore, but I like the way you've worded this. "Multitasking" is a catchall phrase that encompasses a number of different processes. Many people have a very narrow view of what multitasking is and will tell you that you can't multitask on a C=64, and even that you can't multitask on a 386. Many more will tell you that the 386 isn't built to do "true" multitasking (whatever that's supposed to be) and that its multitasking functions are simulated by the operating system (Windows, OS/2, Unix, etc.). Some will even tell you that the 386 is just a "very fast task-switcher" and that you can't do multitasking on a single-processor computer. These are *usually* narrow viewpoints expressed by people with limited experience with only personal computers.

Remember that the term "multitask" goes right back to the late '50's when huge mainframes were allowing multiple users to run different programs from their terminals (not counting "ghost" tasks). Some of the above people that say that you can't multitask on a 6502 will also agree that these mainframes did "true"

multitasking. The fact of the matter is that the 6502 not only has access to *much* more memory than these old mainframes had, but it is also several magnitudes more powerful as well!

So why do people make the claim that the 6502, and even the 386, don't do "true" multitasking? Ignorance, mostly. Some people believe that you must have more than one processor to run more than one program at a time. Not true. The computers from the '50's and most mainframes up through the '80's had only a single CPU.

Sometimes people simply have a misconception. Have you ever seen pictures of those mainframe systems from the '60's and 70's? Row after row of tall cabinets with all the cables running under a false floor? Well, those are "racks" which hold "rack-mount" components. Most of them are disk stacks (disk drives), tape drives, routers, "modems", multiplexers, power supplies, I/O boards, etc. But some of those racks actually hold computers. Each of these computers had a finite capacity and could really only run a dozen or so tasks (we called them "jobs") efficiently. One of the computers would track which of the other computers were running what tasks and would "assign" new tasks to the computers with the fewest number of active tasks. From this setup some people have gotten the misconception that you need more than one computer in order to multitask.

If you are using two CPU's to run two different tasks then you are "multiprocessing". Multiprocessing is a *form* of multitasking! Parallel processing is a method where you have multiple CPUs running different parts of the same program. Parallel processing isn't multitasking in itself, but parallel-processing computers can be made to run independent programs on each processor, thereby multiprocessing, and therefore multitasking.

What you called "time-sharing" above is what we normally call "time-slicing". What happens is that one program gets a certain number of "clocks", after which it must relinquish control to another program (wedge, background music, etc.). Time slicing is a *form* of multitasking!

Some people inaccurately refer to "time-slicing" as "task-switching." Task switching is *not* the same thing. When you "task-switch" you're actually putting your current task to "sleep," and awakening another task. In a task-switching system only one program is active at a time and the "sleeping" tasks cannot do anything. In a "time-slicing" multitasking type of system there can be things happening in two programs at the same time (downloading, formatting disks, background music, etc.).

Now, some processors multitask better than others. Those that are designed specifically for multitasking, such as the LSI-11, to it very well. Some, like the 6502 or 8080, don't multitask particularly well at all. Some, like the Z-80 with its alternate register set can be *made* to multitask reasonably well, but still isn't really designed for it.

Most people will tell you that the 8086, with its small register set and small addressing space, wasn't designed to multitask. Garbage! Yes, it does have only a few registers, but the 8086 was designed for multitasking mainframe and mini-mainframe computer systems. At a time when most mainframes still had 4k-16k of dynamic Ram (like my Geac System 8200 or my DEC PDP-11/03), the 8086 was able to address 1-Mb, with each "task" having up to 64-kb of memory to itself! It may not have been designed to multitask by itself, but in '83 Intel was still marketing an iAPX 86/30 processor, which was the 8086 with an 80130 operating-system coprocessor. This worked just like their math coprocessor, except that instead of processing floating-point and transcendental math instructions it processed multitasking kernal "primitives" (commands) such as Sleep, Delete Task, Suspend Task, Create Segment, Signal Interrupt, Create Mailbox, Get Task Tokens, etc., etc., etc.! (Just for information's sake, the 8086 and 8087 math coprocessor set was actually sold as the "iAPX 86/20 Numeric Data Processor".)

So while it's true that a single processor can't execute two machine instructions *simultaneously*, you can still have two *tasks* active and running at the same time; thus, multitasking. The 6502 doesn't multitask well, but you can get it to multitask.

PPP on the C64?

By Robin Harbron. PPP stands for Point to Point Protocol - it's become the standard way for a computer to hook up to the Internet through a modem. PPP is available for the Amiga, *NIX, Macintosh and Windows platforms, but how about our favorite machine? Is it possible? And why do it?

In my opinion, the #1 reason to implement PPP on the C64 is to allow many more people on the Internet. In some regions of the world, it seems easy to get a shell account, which allows one to use (but not truly be "on") the Internet. In my area, however, not a single ISP offers shell accounts! This seems to be a growing trend, as more and more people only want PPP connections, and Internet providers try to tighten security (shell accounts seem more susceptible to abuse). A PPP account is the only way for these people to get on the Internet.

The other important reason is that with a shell account, you're not really on the Internet. You're dialed into a computer that is on the Internet. For some, the distinction is unimportant, or irrelevant. But what it means is that your computer takes on the bulk of the processing involved, but also has direct access to all the information on the Internet. Additionally, your 64 can be reached by others on the Internet - imagine running an FTP server on your 64!

Rather than talk directly about PPP longer, I'll spend the rest of article discussing the other important components that work with PPP to get you on the Internet.

Many of us are familiar with the advantages of owning a modem. A modem allows your computer to talk to other computers - dialing up a friend to upload/download programs, calling a BBS and leaving messages for other people, etc. And it mostly doesn't matter what kind of computer you're calling. A C64 can talk to an Amiga or Macintosh over the modem fine. The two modems and computers have various protocols that they share in common to allow this. Xmodem, Zmodem, and even ASCII, in a way, are all protocols - protocol meaning "a standard method of communicating".

The Internet is simply many, many computers hooked together using a standard set of protocols, commonly called The TCP/IP Protocol Suite. TCP (Transmission Control Protocol) and IP (Internet Protocol) are the two most fundamental protocols; however, they are not the only ones. Also included are FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), TELNET, PPP, SLIP (Serial Line IP, a precursor to PPP) and many other less famous ones.

IP is the simplest and most important protocol of all. IP offers unreliable, connectionless service. That doesn't sound too impressive, but it's the very heart of the Internet. Simply put, you give IP some information (e.g. a few bytes saying "hello") and a destination address, and it does the work of making an IP datagram which is then sent on the Internet. Why is it unreliable? Because parts of the Internet

(Continued on page 7)

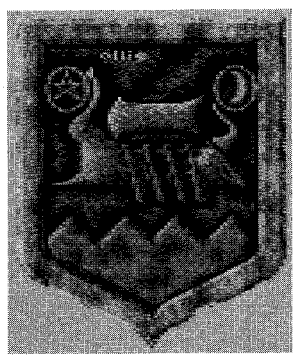
WEB WATCH!

http://www.geocities.com/SoHo/Studios/6463/

A Brief Description Of C-64 Graphic Modes

I found the C-64 Art Gallery on a link from the GoDot page. While you need a graphical browser or C-64 conversion program to see these images, the pages do a good job in paying homage to the C-64. I wonder what PC and MAC people think when they stumble on *these* pages. Sure, there's no hard photo-realism here. No 24-bit graphics, but tons of style.

MCI Picture By Tyrant/Therapy



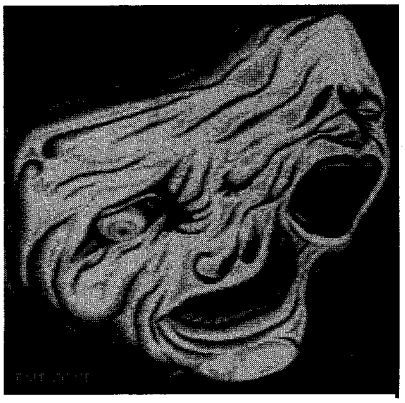
multicolor picture by Ollie/Pride

There are also more links on this page than you can shake a stick at. Take a look at this informative article by the web page creators. English doesn't appear to be their first language. I've made a few corrections. Any error is probably my bad. ...Jeff

HiRes mode generates a screen with resolution 320*200 pixels/16 colors. The screen is divided into 40*25 attribute cells. Each attribute cell is 8*8 pixels big. In this mode you can use maximum 2 colors in one attribute cell. It means that you can set color of background, and color of foreground (pixel color).

This mode isn't very popular although it's possible to paint very nice pictures in it. Nowadays artists use this mode but they almost never use more than 2 colors in it, probably because it's a pain to play with these tricky attributes.

MCI Mode MultiColor Interlace generates 320*200 pixel resolution. You can use 4 colors in each 8*8 points big attribute cell. This mode also gives the possibility of mixing two colors together so there are 4 of theoretical amount of 128 colors. But how it is realized?



MCI Picture By Tyrant/Therapy

IFLI picture by Valsary/Samar/Lepsi

Here is the answer: This mode uses two multicolor pictures (160*200/4 colors in 4*8 cell), each using its own attributes (except color RAM at $d800, which is shared by both). These two multicolor pictures alternates each frame, but one of them is shifted one point to the side. This causes visible effect of jerky movement from side to side. Fortunately, it also improves the fineness of C64 multicolor mode, and gives painter larger color palette.

FLI picture by Electric/Extend (Morbid Art I

This mode is used in pictures "Manga" Tyrant/ Therapy, "Herb's Shot" /Electric/ Extend, and some others.

IFLI Mode, Interlaced Flexible Line Interpretation, is the connection of two greatest ideas in history of C64 graphic modes — MultiColor Interlaced mode (MCI) and Flexible Line Interpretation mode (FLI). Interlaced FLI generates 320*200 dots resolution using 6 (Am I Right?) colors in each 8*1 points big attribute cell. This mode also gives the possibility of mixing two colors together so it's possible to use theoretically 128 different colors. In this mode two FLI pictures alternates each frame, but one of them is shifted one point to the side. This causes visible

Examples of SuperHiRes by Latifah & Dickens

effect of jerky movement from side to side. Fortunately, it also improves the fineness of C64 FLI mode.

MultiColor mode generates screen with resolution 160*200 pixels/16 colors. One pixel represents an area 2*1 pixels big. The screen is divided into 40*25 attribute cells. Each attribute cell is 4*8 pixels big. In this mode you can use maximum 4 colors in the attribute cell. Each pixel defined by 2 bites can have one of 4 different colors. One of colors is background color. As you can observe from today's competitions and demos, almost nobody is able to paint nice pictures in this mode. Artists use MCI, FLI, IFLI, SHI, and SHIF modes instead. Though it's a bit demagogic saying that, ☺ because as you can notice from X-97 gfx compo. Bundy/WOW won competition with cool MultiColor pic. Maybe it's exception proving the rule, maybe just sign that WOW has one of the best multicolor artist of nowadays ☺.

FLI —Flexible Line Interpretation mode generates pictures in resolution of 160*200 pixels. This technique is used for adding more colors into attribute cell. It allows artists to use full palette of 16 colors in each 4*8 pixel big attribute cell. It's realized by very special routine, which by changing values in the register $D011 causes, that Bad Scan Line is on each raster line of visible screen.

For those who don't know what "Bad Scan Line is: Bad scan lines occur each 8th raster line. It's the time when the graphical processor of c64 (VIC) is loading graphic data from memory. In FLI, a "Bad Scan Line" happens on each raster line of the graphic screen. This means that VIC reads the values from attribute screens each raster line, not only each 8th as it normally does. Then, when you change the $d018 register on each raster line, in order to change attribute screen, it is done. You can use 4 colors in each 8*1 pixels big attribute area.

Roland Toegel - Crossbow/Crest - is the inventor of the SHF and SHIF modes. For the few who don't know, let's say that he is famous coder and excellent (YES!) artist. He is in the scene since eighties, and since the beginning he together with Goldrush, produced demos based upon excellent ideas. He is cool, sympathetic, friendly, cheerful, and definitely NOT arrogantly jealous to the new faces in the scene as some self-established old-legends. Like some other famous and still enthusiastic dudes, I had chance to virtually speak with, he does *not* act like pathetic self-admiring ego which consider words lamer and newcomer as synonym. He didn't give me the condescending, and patronizing advice on "how to approach him" even though he never heard of me and didn't see any product of Studio Style. Demos he released lately shows us all the right attitude, sharply contrasting with all these badly designed PC demo clones. To Crossbow and Deekay you can thank for the good colors in all new pictures in the gallery.

If you want to see some nice pictures in SHF, SHIF mode, and even more, check demo Krestology in the Heroes and Legends chapter on this web page.

Super Hires Interlace generates pictures 96*200 pixels big.

It alternates two SuperHires pictures each frame.

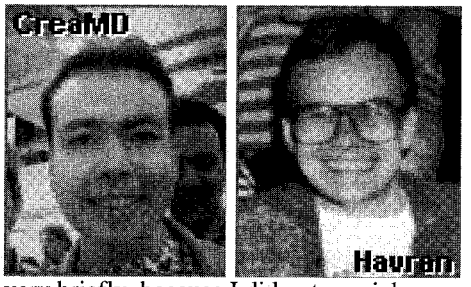
Like all interlacing modes, it can theoretically generate 128 colors palette.

I think that now it's time to check the example picture. For me it was shock when I saw this picture for first time. The picture is painted by now inactive polish painter - Latifah.

Believe it, or not, converted to non-

interlaced mode it has 87 colors and in one 8*8 pixels big attribute area it can use 10 colors.

I'll only list other graphic modes there very briefly, because I did not special converter for them. They are HiRes FLI, SHI, SHII, SHF, SHIF, UFLI. Except Super Hires, these modes are unemulatable. As I don't have done converter for these GFX modes, I'm too lazy to learn exact parameters of each of them, and therefore I'd rather not write any info about them. GET THE REAL C64 and CHECK THEM!

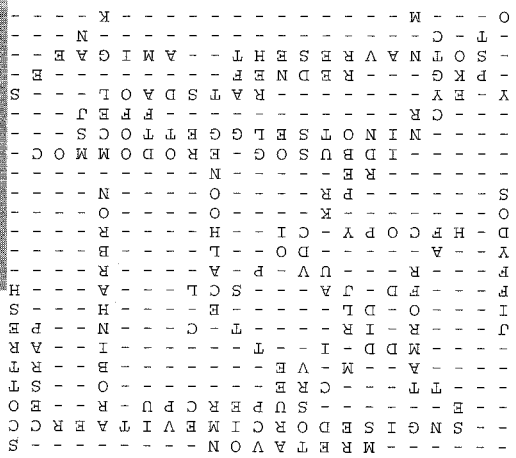


We, CreaMD and Havran, decided to place IFLI gallery on Geocities because the previous host on Slovak Academical Net. wasn't very fast. So, we moved into SoHo/Studios neighborhood, where we got 2-megs of disk space for free. 2 megabytes is not that much but anyway, it's for free, and that's what student with minimal income can appreciate. Therefore we thank to www.geocities.com for their free www homepage program!

C64 Picture gallery is full of cute little pictures, drawn on Commodore 64 computer. Pictures are converted from various graphics formats mainly HiRes, Multicolor, IFLI, FLI and DRAZLACE.

mailto:rchlebec@decon.unitra.sk

SOLUTION TO PUZZLE ON BACK



(Continued from page 4)

can get too congested, machines can break, and your datagram is lost as a result. Also, one packet can manage to get ahead of another, and can arrive out of order. Connectionless? If you send someone a letter, that individual letter doesn't allow a conversation to take place directly - each letter is just a stand-alone, one way communication. This is what IP is like.

TCP on the other hand, uses IP to provide a reliable connection over the Internet. TCP receives a stream of information, which it then breaks into chunks, and sends them in numbered IP packets. TCP at the receiving end makes sure that the IP packets come in order, and that none are missing - if there is a problem, it tells the sending TCP to retransmit as appropriate. This is akin to sending many letters to someone, and them writing back - making sure each letter is numbered so that you can be sure to read them in the correct order, and know that you haven't missed any.

Now, obviously every computer on the Internet can't be directly hooked up to every other computer. It's fine when you want just one computer to hook up to another - but how do you handle it when 3 computers want to hook up at the same time? How about hundreds -- even millions? There are two main parts to the solution to this:

Every machine on the Internet is assigned what is called an IP address. The addresses look like this: 206.212.31.12. The address is simply 4 bytes separated by three "dots". To the computer, it's just a 32-bit word. There are protocols that allow more readable names, like www.arkanixlabs.com, for example - but these just get translated into the numeric address.

Each of these machines is hooked up in groups (like all the customers dialed into their Internet Service Provider) to one or more routers (sometimes called gateways) that also have IP addresses. The router simply directs the information that arrives onwards in the appropriate direction. This is commonly called a "hop". There may be many hops before the information reaches its destination.

How will all these protocols and systems work with the 64? Well, the appropriate software will have to be written. I'm working with Arkanix Labs to implement a TCP/IP stack, called Netstack. Why is it called a stack? The way the system works internally is like a stack of protocols, each one talking to its immediate neighbor. I'll explain what will happen inside the 64:

For example, you are running TELNET on your 64 (this is very similar to a terminal program like Novaterm, except that it works directly on the Internet). You want to connect to another computer somewhere else on the Internet. You instruct TELNET to open a connection. TELNET passes the open command on to TCP. The TCP program then tells IP to send a packet (or possibly multiple ones, if the command is large). Your IP program then sends the packet (or packets) on to PPP (or the simpler, but still capable SLIP). PPP then feeds the information out over your modem, which is received by PPP running on your ISP's machine. PPP then passes that information on to IP running on your ISP's machine, which then decides where this packet should go, and then sends it on using the correct protocol for that link (it could be a Novell network, for example). This process continues until the destination machine receives it, when it is passed from IP up to TCP up to TELNET. The TELNET application then sends back the appropriate response (error or acknowledgment), and on it goes, sending and receiving packets in this way.

There are many choices in how to implement this, but a relatively simple stack is well within the reach of the C64, as proven by Slipdemo. And just wait to see what the SuperCPU will be able to do! Email me if you have questions about Netstack: robinh@arkanixlabs.com.

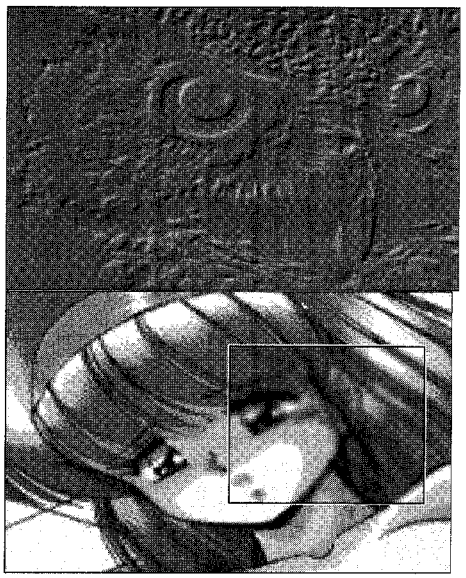
GoDot Long Overdue In The US

By Robert Bernardo. Foreword from Jeff: Godot is a German product, not yet available in the US. In a breath, GoDot allows you to manipulate your C-64 images. We have the full version and will hand it over to Walt Harned for his review. The demo is available now in .d64 format at

http://members.aol.com/howtogodot/welcome.htm

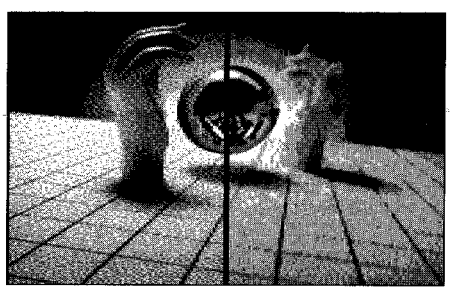
and will be available on the next Star extra disk. After Robert's testimonial, I'll give a brief listing of the features, taken from the manual.

Robert: GoDot is a program that is a long time in coming to the states. Ever since Commodore World magazine mentioned it in its defunct column, Foreign Exchange, many months ago, I've been waiting to get my hands on it. A few months after that column, the first



GoDot demo appeared in the Genie libraries. Eagerly, I downloaded it, and though many of its features were disabled in the demo, it was enough to convince me that this was the program to have.

I tested it with a multicolor, Koala pic of The Terminator. I threw every kind

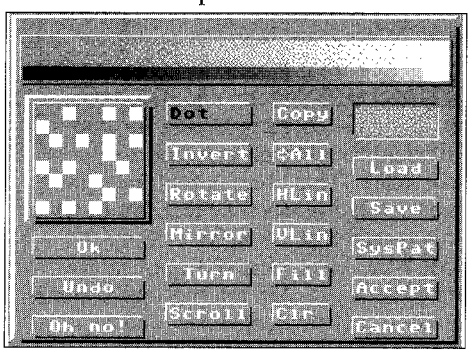


of GoDot effect at Arnie. By the time I was finished with him, he was literally a melting mass of a picture, not at all like his former, crisp, Koala self.

I was impressed! Here was a program that had never been done before for the Commodore--a color, graphics processing program akin to those on Amigas, Mac's, and PC's. ...And the demo was just a taste of what GoDot could do, because the full version would handle more picture formats and do file translation.

So impressed was I with the program that I distributed GoDot demo disks to interested members of the Fresno Commodore User Group. They even asked for a live demonstration of the program, but I held back, because I didn't have the full version.

I thought, "No problem." I had read on comp.sys.cbm that Creative Micro Designs was to handle GoDot distribution in the states. I sent them an e-mail letter last year but received no reply. Late last year, at an on-line conference with Doug Cotton, I asked him directly when GoDot would be ready for sale. He said December. He said the English translation of the manual was nearly done. Nothing happened. In May of this year I pinned him down again at another on-line conference, asking when GoDot would be ready. He said that the manual still needed more cleaning-up on the translation. He said the beginning of June. Again nothing showed up in any CMD advertising. I directly called CMD at the beginning of June and asked about GoDot. The lady who took my call paused as she asked someone and finally got back to me by saying, "GoDot is not in our immediate plans."



Well, I couldn't immediately follow up on this situation, because I was leaving for England on June 10. When I arrived in England, my access to e-mail was

limited, and I couldn't communicate to Arndt about what had happened. (With each Cotton conference or CMD experience, I kept him apprised about what was going on. To my surprise, he was more in the dark than I was. He hadn't heard from CMD in over a year!) While in England, I did have immediate access to Commodore public domain companies and programmers that I had heard of. If I needed a certain disk or if I wanted to talk about their projects, they were an easy phone call away. They were most eager to talk to this American.

When I returned to the US on July 8, I had learned an important lesson from my British experiences. Being direct is the best. If I wanted a certain program, I should go directly to the author. That is when I contacted Arndt and asked him whether I could directly order GoDot from him. I reassured him that we had someone who could do a rough translation of the manual. He gave me the e-mail address of Performance Peripherals--Europe and suggested that I try to order GoDot through them. No reply--I guess they didn't understand my English.

In his last e-mail to me, he bemoaned the fact that he couldn't find a reliable distributor in the US and that all of the potential American users were being left out. I remembered that in a Delphi conference with Gaelyne Gasson she had mentioned her book, the Internet for Commodore Users, was being distributed by LOADSTAR. In a follow-up phone conversation, she confirmed this fact. I suggested to Arndt to contact Gaelyne and LOADSTAR for help.

In summary, GoDot is a fine, powerful program in which a number of picture formats can be utilized and a number of effects can be applied to a graphic. I cannot understand why there is a delay in distributing this program in the states. Europeans have known about this program for some time. It's time Americans get a chance to use it, too.
Robert Bernardo is president of the Fresno Commodore User Group and a member of the Stockton Commodore User Group.

Jeff here again: I was impressed, too. It supports the Handyscanner, video digitizer and up to 720-dpi printing. The web page shows images captured with GoDot and they look a lot better that what we're used to in America because GoDot supports the fancy screen modes discussed on pages 5-6.

I have an Amiga, but no graphics

tools in the price range with true filters anywhere near as useful as GoDot. GoDot reminds me of the filter menu of my PC's Photoshop. Filters algorithmically change images. I tried a few of Walt's images and was astonished that GoDot's filters do what they say. Here's a brief and *partial* list of GoDot's filter modules.

- Blur High - Produces a heavy blurring effect.
- Blur Low - Guess!
- Cartoon - All colored areas are cleaned up and surrounded with a black border.
- CleanUp - Removes a pixel if none of the neighbors has the same color.
- Convolve - This module acts as general-purpose convolution filter, which can apply 3x3 convolution matrices to the 4-Bit data. A number of presets are available, your own creations may be saved to disk.
- Convolve7-Bit - Use this tool to apply convolutions to image data in TED 7-bit format imported via the PCXprep4Pl4 loader.
- CrossHatch - Applies a fine texture to the picture.
- DeepPress - All objects look like they are pressed into metal.
- Emboss - A very popular relief effect, changes every picture to the surface of a coin.
- Gaussian - Less blurring than the special blur operators.
- LaPlacian - All areas loose their colors, only the borders are left.
- MedianFilter - A useful tool to eliminate unwanted noise in digitized images.
- MedianFilter - Performs the process on 7-bit TED data imported by the PCX-prep4Pl4 loader.
- MosaikMaxi - Replaces a pixel with the brightest color of the neighborhood.
- MotionBlur - Uses the width value of the current clip to set the amount of smearing the colors to the left and to the right.
- OutlineColor - Similar to mod.LaPlacian, but with colored borders. The less details, the better the result.
- OutlineMono - Similar to mod.LaPlacian, but with white borders. Use it in conjunction with mod.Negative to produce mask images.
- RandMoveLeft - Moves the pixels to the left up to a specified distance. If used several times the whole image is blown out of the screen.
- RandOblique - Smears the colors in diagonal direction, thus simulating a brush effect.
- RemIsoPixels - Replaces a pixel if all neighbors are of the same color.
- Sharpen1 - Puts more accents on borders to enhance sharpness.
- Smear - Randomly swaps pixel values.
- Speckle - Similar to mod. Smear but with more significant results.
- WaterColor - Simulates the use of water colors, preferring the most repeated color of the surroundings. The Oblique version adds a diagonal brush effect.

REUs, VICs & Super Snapshot

By Robin Harbron. Last month, while trying to write a SuperCPU routine, I needed some hard facts about how the VIC works. I wanted exact timing information about it. Now how does one do this without doing all sorts of complex hardware analysis? Easy – use the REU.

The REU is such a powerful beast - it goes far beyond just memory storage and retrieval. It has the ability to record the contents of a particular memory location in the C64, as many as 65,536 times in a row. This is of no use in a non-changing location, like somewhere in RAM. However, some of the hardware registers change on a cycle by cycle basis. To find out more about how the VIC works, we'll center in on one particular location - $D012.

When $D012 is read, it tells which screen line is currently being drawn. More accurately, it gives the low 8 bits of the current raster line number - the 9th bit is located in bit 7 of $D011.

How is the screen drawn? Your TV or monitor starts drawing in the upper left corner of the screen, then works its way to the right edge of the screen. Then it drops down one scan line and draws across again. The whole screen is redrawn 60 times a second on NTSC (North American) machines, 50 times a second on PAL (pretty much the rest of the world) machines.

How many scan lines are there per frame (screen redraw)? It depends on what sort of VIC you have in your computer. Mapping the Commodore 64 says there are 312 (horizontal) lines in the PAL standard, which are numbered 0-311. The model number of the PAL VIC is 6569. Mapping also says that there are 262 lines in the NTSC standard. This is only true for the odd 6567R56A model VIC, which happened to be common when Mapping was first written. Apparently before, and certainly after, that revision of the 6567 was a 263-line model, which is far more common and popular (with demo-making people) as well.

Knowing this information puts us in a good position to learn more about the VIC. I've written a short program which uses the REU to scan $D012 on a cycle by cycle basis - we'll analyze the data afterward. You certainly don't have to run this on your own, but you may find it fun, if you're like me at all.

```
*=$8000
lda  #%10000000
sta $df0a ; fix C64 address
lda  #<$d012
sta $df02
lda  #>$d012
sta $df03
lda #0
sta $df04
sta $df05
lda #2
sta $df06 ; bank 2 of REU
lda #0
sta $df07
sta $df08 ; 65536 bytes
lda  #%10010000
sta $df01 ; do the transfer
rts
```

This program sets the REU to look at $D012 in the 64, and copy 65536 bytes from $D012 to bank 2 of the REU. My assembler happens to use bank 0 and bank 1 extensively - feel free to use any bank that works for you. After running the program, we're left with a lot of numbers in the REU.

I use my Super Snapshot to look around in my REU - I've heard the much-hyped Action Replay can't do this (just a dig at my AR-using friends). Just press the Snapshot button, hit M to shortcut into the ML monitor, type *R2 (return) to look at REU bank 2 memory, F1 to go to the bottom of the screen, M 0 (return) to look at memory starting at 0, and then cursor down through the data.

I didn't have a PAL machine to test this on, so all following info is only regarding the NTSC VICs.

What do we see? The contents of $D012, on a cycle by cycle basis. So you'll see one value repeated sixty-something times, then that value increased by one, repeated again, and so on. There are a few curious things to note:

Remember that we're only looking at 8 out of the 9 bits. This means that when we go past $FF, we'll wrap back to 0, but in fact, it's $100, as the high bit is missing. Then it will count up to $06 (263 lines total, this is the more common VIC) or $05 (262 lines in total, the 6567R56A), then go back to 0. This repeats not quite 4 times in the REU bank, meaning we've captured about 4 frames worth of data.

If you have one of the more common VICs, you'll find that most of the scan line numbers repeat 65 times. The odd 6567R56A has 64 cycles per line. There are 3 exceptions to this rule:

The first scan line, number 0 (not $100, which shows up as 0 because we're missing the 9th bit) has one less cycle than the norm. So this means 64 or 63 cycles depending on the revision VIC you have.

The last scan line, either $106, or $105, again depending on VIC revision, has one extra cycle. This will be either 66 or 65 cycles.

Every 8th scan line, starting at line $33, and ending at line $F3, 43 cycles go missing. Depending on your VIC revision, either 22 or 21 cycles are on these lines.

Exceptions 1 and 2 are a little curious, but really no big deal - it probably has something to do with the circuitry inside the VIC having to reset it's counter back to 0, instead of just increasing by 1.

However, exception 3 is very strange! What is going on here? I've gone into this in an earlier article, but it may make more sense now.

The VIC is not like a video card found in IBM compatible machines. Those cards have their own memory, which the processor can't directly modify. Likewise, those cards can't access the main memory in the computer - it can only look at its own separate memory. The processor has to feed memory into the video memory byte by byte. The VIC, on the other hand, shares the same memory that the 6510 processor in the 64 uses. Of course, the 6510 can look at all 64K at once, while the VIC can only look at one 16K quarter at a time.

How do the VIC and 6510 get along in this? They take turns each half-cycle. The VIC takes the first half, then the 6510 takes the second half. The VIC needs to grab graphics data every cycle to keep the screen display going. In this half cycle it either grabs font or bitmap data, depending on what mode it is in. It spends 40 of its cycles every line doing this. During the other 25 or 24 cycles each line it grabs things like sprite pointers.

However, at the start of each 8-pixel high character or bitmap line, too much information is needed at once. The VIC has to fetch both the pixel data, and also the character/screen data ($0400, 1024 area, by default). So the VIC "stuns" the 6510 temporarily, and steals its cycles. When the VIC stuns the 6510, it also halts the REU. In fact, when you are executing a REU command, the REU stuns the 6510 itself.

Lines on which the VIC steals extra cycles are commonly called "bad lines", because the VIC makes the 6510 unproductive for a time. So the scan lines that are recorded as being 22/21 cycles long are in fact still 65/64 cycles long. The REU is just not able to read memory during this time.

Why do 43 cycles go missing? The last 40 are used by the VIC to get the screen data. The first 3 are a grace period given to the 6510 to finish off any work it is doing before being interrupted.

Additional cycles will go missing when sprites are enabled. I haven't fully studied this yet, so I don't have specifics, but be aware of it. 4 half cycles per sprite sounds like a reasonable number - 1 to fetch the sprite pointer and 3 more to fetch the 3 bytes of graphics data per scan line per sprite.

This is only a small step into the inner workings of the VIC. There is so much to learn! Knowledge of bad lines is the key to creating many of the demo effects that are seen on the C64. Bad lines can actually be made at will on the C64, by knowing the right trick. For example, a software graphics mode called FLI actually forces a bad line on every line - this causes the colors to be changed on every scan line, allowing you to have all 16 colors that the C64 can display in one 8x8 pixel area! I'll discuss this in a future article.

In summary, here's a chart detailing our discoveries:

	CYCLES		
SCANLINE	6567R8		6567R56A
$0	64		63
$1-$32	65		64
$33	22	*	21
$34-$3A	65	*	64
$3B	22	*	21
$3C-$42	65	*	64
.			
.			
$F3	22	*	21
$F4-$FA	65	*	64
$FB-$104	65		64
$105	65		65
$106	66		NA

* denotes visible screen

File Manipulation On CMD Devices

By Jeff Jones. There's a lot of power in a disk command, particularly on a CMD device. If you have your CMD drive partitioned, it's easy to move a file from one partition to another or scratch a file in a partition that's not current. All you have to do is understand how your CMD unit is laid out, and what it can do.

The drive number/partition number: Y'know how some people feel it mandatory to use a drive number when dealing with files? Consider the following:

```
save "0:filename",8
open2,8,2, "0:file,p,w"
@r0:newfile=old file
```

In each case the "0:" is the drive number, which used to be a mere throwback to the days of old dual drives, and quite superfluous on a 1541 since there was no drive 1 to refer to. There were exceptions for the use of drive numbers. First, using a drive number with the save-with-replace syntax does avoid a bug in the 1541 OS. The other exception is that the drive frees up buffer space (I faintly remember) if you use a drive number when opening files. Other than that, I have actively avoided using drive numbers for ten years now with no problem.

CMD came along with their HD-DOS, RL-DOS and FD-DOS and suddenly I found a need for partition numbers. I have my RAMLink partitioned into seven virtual drives, and 75 partitions on my HD-85. Now the partition number comes in handy when I'm manipulating files on my system.

Copying files: I keep all my utilities, fonts and some source in partition two of my HD and RAMLink. So when I need the standard LOADSTAR font in a program I'm working on (usually in partition ten), I just copy the font to partition ten. For this, I don't need FCOPY. I just issue the following command:

```
@c:font=2:font
```

"2:font" is the pathname of the font file in partition 2, and is the simplest type of pathname you'll use. Note that I didn't specify that the file should go to partition 10 because partition 10 is the current directory, but if I want to be verbose:

```
@c10:font=2:font
```

Note that the colon always immediately precedes the filename. For instance if I were copying from a subdirectory called binaries in partition 2, the command would look like this.

```
@c:font=2//binaries/:font
```

"2//binaries/:font" is the pathname of the file you're creating. Note that I follow the partition number with two slashes. This makes the path start back at the root directory. This is necessary in case the current directory isn't the root, which would make the path invisible. For instance if the current directory in partition 2 is already *binaries*, and it tries to find the subdirectory, *binaries*, it won't find it. And if I wanted to copy it from

that subdirectory to a particular subdirectory in partition 72, it would look like this:

@c72//my program/:font=2//binaries/:font

You don't have to have partition 2 or partition 72 as the current partition. You don't even have to have the device as the current JiffyDOS device. If the hard drive is device 12, and your commands would normally go to device 8. There'' no need to press CTRL-D or @#12. You can place your command within quotes and give a device number:

@ "c72//my program/:font=2//binaries/:font",12

Booting programs: This is the same. To boot a program in the current directory, you can type ↑program. If you must, you can use ↑0:program. TO boot a program in partition 13, type ↑13:program. If you want to boot a program in a nested subdirectory in partition 69, type *↑69//binaries/games/adventure/:mst3k.sda* and again notice here that the colon precedes the filename, no matter how complex the path. *MST3K.SDA* is the program we want to run and 69 *//binaries/games/adventure/* is where we find it. There's only one caveat here. Keep in mind that most programs aren't designed to boot from a particular directory. They boot from the current directory, whatever it is. So if you try to boot LOADSTAR in partition 6, but you're really in partition 1, the LOADSTAR in partition 6 will try to find its files in partition 1 unless you alter the boot file to change the current partition with @cp6 before the program tried to load any files.

Viewing a directory: is usually accomplished with the JiffyDOS command, @$0:* but we can view any directory or subdirectory without having to move to it first. To look at partition 2 we'd use: @$2:* and to look in the binaries directory there, we'd just include the path @$2//binaries/:*

All disk commands and procedures such as save, validate, scratch and even rename work with partition numbers and pathnames and the way you know they work is by checking the error channel as normal. The number one rule to remember is that the colon precedes the filename, even if it seems like it belongs closer to the partition number.

Stupid PET Tricks Part Two

By Scott Elliott. This is continued from the previous issue.

PEP UP THAT ML SUBROUTINE WITH THE SEI AND CLI INSTRUCTIONS! Consider this: Interrupts happen sixty times a second, and they do a lot- They update the screen, check the keyboard, service the timer and generally wreak havoc. If you have a ML subroutine that isn't just up to par, put the SEI instruction before it, and when the subroutine is done, insert the CLI instruction to restore the interrupts. The theory goes that without the hassle and time consumed that the interrupt requires, your ML subroutine now commands nearly 100% of your microprocessor's time and runs quicker. Of course, a NMI request would still muck up your brilliant piece of compact ML code.

THE JMP ($XXFF) INSTRUCTION: It ain't a bug, it's a feature! I know you may raise an eyebrow or two, but bear with me for a minute. First of all, let me briefly explain what the indirect JuMP instruction does: JMP ($033C) means that the first address ($033C) contains the low byte, for example $00, and the second address ($033D) contains the high byte, for example $C0. When the 65xx or 8502 microprocessor fetches these two bytes and forms an address ($C000), it jumps to that address. So far, so good. What if it was JMP ($03FF)? Then we have a problem. The 65xx microprocessor gets the low byte from the $03FF address, but due to some quirk, gets the high byte from the $0300 address, not the $0400 address as it should do. This has to do with the 6502's awkward handling of page boundaries, i.e., moving from $03ff to $0400.

Now, what to do? Easy... Just put the low byte into the $03FF, and the high byte in the $0300 address, and then the JMP ($03FF) instruction will now work. It since has been fixed in Rockwell's 65C02 chipset. Since the page boundary is not crossed, the instruction always takes 5 cycles. Why do it, you ask? No real reason other than to muddle up that fine piece of elegant code. Seriously, you could try this technique to verify the presence of an emulator. While I do not have access to the latest gee-whiz c64 emulators, I am willing to bet that they do not emulate the JMP ($xxFF) instruction correctly. Or even the faulty decimal opcodes too.

THE MAVERICK BIT: We have Nine-Bit computers, not Eight-Bit computers! Technically speaking, yes, this may be true. In normal 65xx architecture, we have 8-bits traveling around at any given time. But not a lot of attention is paid to the Carry bit register that is normally affected by addition, subtraction, rotate and shift opcodes. This bit can be easily manipulated by these opcodes, and we can use it to our advantage. With this technique, we can count up to 512 different possibilities that a single nine-bit field occupies, and this flexibility can be an asset.

One good application for this would be compression. For a detailed explanation on how to use the maverick bit, look up the monitor decompress routine in the C128 starting at $b6a1. Another application is for joystick routines. Just read the joystick byte, and LSR or ASL, and read the status of the carry flag to determine what action to take, and repeat, etc. It's that simple. No more ANDing or ORing the joystick byte as you would in BASIC.

Which leads us to a tip you can use by forcing the 9th bit to do some heavy duty lifting. You need to shift bits left (multiplication) or right (division) on a word (16-bit) field, but that word field is in low-byte/high-byte 6502 order. This can be easily done and still preserve the integrity of the lo/hi byte order. The code is:

```
temp =* (word field.)
; assume that temp contains an
8-bit value and we want to
multiply it
; by 16.
multi6 =*+2
 ldx #$02
- asl temp; TEMP contains the
low byte, and we've multiplied
it by a
; factor of two.
 rol temp+1; TEMP+1 contains
the high byte, and with the 9th
bit at
 dex ; work, it determines its
contents.
 bpl -
```

8-BIT ADDITION TO A 16-BIT WORD: When I talk about 8-bit addition to a two-byte integer, I mean this routine:

```
CLC
LDA LOC
ADC #40
STA LOC; low byte
LDA LOC+1
ADC #$00
STA LOC+1; hi byte
```

By adding a zero with the carry flag, (see how important the 9th bit is, eh?) we account for any overflow and update the hi-byte accordingly. This is the majority of the code I've seen in which a single byte was added/subtracted to a two-byte word. This is inefficient. The above routine would be run every time it is called and consumes 22 clock cycles and 17 bytes. (This is assuming that absolute addressing is used.) But compare with the following routine:

```
CLC
LDA LOC
```

```
  ADC #40
  STA LOC; low byte
  BCC +
  INC LOC+1; otherwise incre-
ment the high byte.
+ rest of code
```

The routine, 6 out of 7 times, would not run the INC instruction. This results in only 14 clock cycles consumed. Even with the INC instruction, the routine consumes 20 clock cycles. Not only it is faster than the ADC #$00 instruction, it is shorter as well, taking up space at mere 14 bytes.

This is much more efficient use of the addition routine and is quicker overall. The efficiency decreases if the addition value increases. The same holds true for subtraction, except that you BCS + and use SBC instruction, etc. Called once or twice, the first routine would be okay, but if this routine is to be called repeatedly, the second routine will give you a speed boost. This tip is ineffective in two-byte addition/subtraction routines.

Note: The optimized routine as shown earlier will not reset the carry flag. While not needed for a stand-alone math routine, please keep this in mind when adding more math routines following the optimized routine. For example- To update the screen memory and color memory at the same time, the optimized code shown below will reset the carry flag.

```
   clc
   lda colorram
   adc #40; add a screen row
   sta colorram
   bcc +
   inc colorram+1
+ clc; important- reset the
carry flag by clearing it
   lda videoram
   adc #40; add a screen row
   sta videoram
   bcc +
   inc videoram+1
+ [code continues...]
```

The SEC instruction would be used for those routines involving subtraction. I found this oversight the hard way, using no CLC instruction, and encountering undesired results. I looked the code over, and said, waitaminit! A simple fix there.

Postscript: I was going to put the undocumented opcodes here, but I felt that they were covered adequately in numerous other publications. Additionally, these undocumented opcodes will be wholly incompatible with the emulation mode of the SuperCPU. Rather, I would hope that these tips have helped somewhat for you to gain a deeper understanding of just exactly how the C= 8-bit machines work. Most tips are very esoteric in nature and I would never intend for you to use most of them in actual programming situations. They only serve to illustrate the inner workings of the 65xx and 8502 CPU, and serves to demystify the ML programming aspect by showing its lovable quirks.

For further study in this vein, these WWW Internet links provide rich resources for the aspiring and the hard-core ML programmer alike. For an extensive study of machine language itself, look no further than the 64DOC file maintained by Marko Makela at:

http://www.hut.fi/~msmakela/cbm/emul/x64/64doc.html

For quick and dirty ML routines that you can easily incorporate into your programming, or just want to learn ML by studying short subroutines, look no further than the refrigerator, the Fridge, that is, as maintained by Stephen Judd. Some of my work is there. The URL is:

http://stratus.esam.nwu.edu/~judd/fridge/

Now, armed with this knowledge, I bid you good luck in your ML endeavors.

Error vs. Fact: You Can't Believe Everything You Read

By Jeff Jones. The Internet and most every BBS is full of quick facts that you can't rely on. Just because it's written in pixels does not make it true. Here's a list of false statements and their corrections online and perhaps unfairly off-line by lurker, me.

Error: Programs run faster from ROM. I know because of how fast my terminal program runs in Super Snapshot. If GEOS were in a ROM Cartridge, it would run faster.

The Truth From Jeff: A program on a ROM cartridge such as Super Snapshot doesn't run any faster than any other ML program. The program is only swapped into memory from the cartridge ROM, which is a pretty swift process since no program on the Super Snapshot cartridge is longer than 8K (if I remember correctly). I was commissioned to write an 8K program for the next generation Snapshot cartridge back before it was canceled. The original memory is preserved in the cartridge RAM and swapped back into the computer when you exit the terminal program. This is why it seems as if the program is running from the cartridge. In actuality it's a machine language program running in the computer. It's fast because it's machine language, not because it's ROM. Now a system of programs swapping in and out from a system of ROM or RAM chips would seem faster than a mega-program like GEOS, but no part of the program would run faster. It would only copy routine into memory faster. Still, a mega-program in an REU would swap itself in and out much faster.

Error: The SuperCPU is incompatible with 50% of all Commodore Software.

The Truth From Jeff: This is absurd. This assertion came from a person who doesn't own the cartridge. Software which isn't compatible with the cartridge is usually packed with a program like The Bit Imploder, which uses illegal undocumented opcodes. If every program were packed with such software, 100% would be incompatible with the SuperCPU. Fortunately most software isn't packed with this packer and similar packers. If you primarily run demos written overseas, packed wwith weird packers, which barely run on a stock C-64, maybe you don't want a SuperCPU.

Error: Most BBSs these days really don't have time limits (or at least they give you like 190 mins)

Fact from Online Respondent: Actually most of the ones I call are very busy and can usually only offer 60 minutes max. The others I call I pay for membership and time. And I do quite a bit on them so every second is valuable.

Fact from Jeff: BBSs were designed to be used by many people. There are only so many hours in a day, and if you only have one line, you limit time. I call a Fidonet BBS where my time limit is less than thirty minutes. I do fine there.

Error on the value of TIFCU: Why spend the money on a book when you could get the same information (if not better) for FREE... then you could buy some stuff for your C64 or C128 (or maybe even add an AMIGA).

Fact from online respondent: Okay, let me ask you this question: *Where*? If there are *no* user groups, *no* stores, *no* neighbors, and *no* knowledge of how to connect a 64/128 to the 'Net, *where* is the average user going to get this info? "Well, gee," you say, "Just log on to my BBS over here and get it free!" Yeah, right. You just gave them a Catch-22 situation: If they can get it from you, they'll need to log on, and to log on, they'll need your info. Great.

Fact from another respondent: Books are a huge necessity in this world. And books about the Internet are in big demand and quite widely available. But books designed mostly for Commodore users on the Internet are rare. Gaelyne has filled that need. Once you have gotten

into a pattern of things that you do everyday (on the Internet) it must becomes second nature. You do the same thing and it works for you. But what if there is something that you don't understand? Do you spend a couple of hours trying to figure it out or searching for some FAQ listing somewhere, and maybe you don't know where to find it? No, dig out a book and read about it. Many people have a hard time just getting to a certain step because they don't understand how to do it. Well-written books are valuable, because you get a chance to read how to do something in someone else's plain English.

I personally have trouble finding the information on the Internet that I want. I have found that UNIX programmers control much of the Internet. UNIX programmers tend to think that people using their systems already know most everything. This is bad thinking. There are millions of people out there that know very little and will continue to know very little until they learn it from someone. The idea of a book being made available about using your Commodore on the Internet, and one that is written by someone who owns and uses a Commodore and is very familiar with the Internet is a very welcome sight to many.

I plan to order this book from Gaelyne soon. I'm sure that it is filled with a great deal of information that I can use to my own benefit as I create tools for use on the Internet.

Please don't knock the book unless you have already seen it and read it. Professional book reviewers always read through a book prior to telling someone whether it is any good or not. Maybe you can tell us exactly, which parts of the book are not to your liking? Please describe the chapters that are not beneficial to any Commodore user. Don't take this message badly. I'm just pointing out that there are many people that can use this book. Your previous messages might tend to scare people away from it.

Error: Who needs more RAM or a SuperCPU? Nobody!

Fact from Maurice Randall: Having the ability to use up to 16-megs of RAM in the SCPU is what will really exploit the power of the thing. The new OS upgrade for GEOS already handles it. The 16-megs can be used entirely for a ramdisk or just portions of it. Any new application can make use of whatever size chunk it might need for itself. The OS will allocate the ram as needed. Being able to manipulate huge graphics entirely in direct access memory is much faster than using the virtual memory of a floppy disk or a hard drive. Just take a look at how slow Windows '95 is with only 8-megs of ram as compared to having 16-megs or 32-megs. Things happen much quicker since that big swap file is smaller or non-existent.

I think that you are only considering the need for the 16-MB with any existing software that is already out there. Forget it. The games that 64 users play work in 64K of ram. The 16-MB are not needed for those games. Nor is it needed for any current productivity software. But that is not the idea behind it. You have to think of the future, not the past.

Error: C64 users have A right to be MAD!!! - when these PC users are running most of the so called C64 FTP sites... they could at least put the files in C64 format and not some PC formats. If I wanted to download files in PC formats - I WOULD BUY A PC! ☹

Facts from online respondent: Who *owns* the hardware the FTP site is on? All I am saying is, if you don't like the way it is, *do* something! Don't cry about it, I am *sure* there are many 64 users that also have a PC, set up an FTP, get the files, use a conversion program on the PC to export them back to 64 friendly files and put em on the 64/128 friendly FTP, you just can't expect a PC only user to set his/her FTP site up to please you, it don't work that way.

Also, you could try emailing the FTP operator, I run a BBS, and I often change things for people who ask NICE :)

A snipe from Jeff: There are a lot of avid C-64 software fans out there. These people use emulators to a great degree. When they stockpile software at their FTP sites, they should keep in mind that there are people running *real* C-64's who have trouble with ZIP files. I downloaded a ton of SID music that I could easily play on my PC, but had a hard time hearing on my C-64. When I told the webmaster about this, I got my head bitten off and was called a *dumb American.*

Error: we wouldn't feel the need to flame "CMD" - if they didn't keep us 64 users in the dark! First they claim that the SuperCPU will bring instant life to the 64 but I don't see much support yet! - The CPU has been released for some time now!

Jonathan Mines tears out a new one: Seems to me that someone here has an ax to grind with CMD. I've followed all these flames toward CMD on this newsgroup for a while, truly pathetic people. Do you care at all about your C64 or C128 machines? Complain about CMD prices? Have you ever tried to run a business? Know anything about the bottom line?

SuperCPU 64/128 is a great product. We at Arkanix Labs are working on products for it. We are waiting on the RamCard release to get our first product out, a game we've imported from Europe. Most of our software will not be games though, we are moving out of that part of "the market."

A press release from Arkanix Labs on Friday will explain everything -- our goals and what we hope to do to support this market in 1997/1998.

Why not give this attack on CMD (and Doug) a rest, it's lame and childish.

Jon Mines - jonm@arkanixlabs.com
Arkanix Labs -> Software & Hardware Developer
http://www.arkanixlabs.com

Error: in a response to a person with a copy problem. Original question: Recently I wanted to copy a data disk from a 1571 drive to a 1581 drive. I used the MCOPY program from the HD utility disk. I tried using the proper "F" keys to select the target and source drive. I tried and tried, and why will it not accept a source drive as a 1571 and a target drive as 1581? I do not understand.

Partially true answer: Your problem is simply this: Mcopy is designed for bulk copying between like disks (copying from a 1581 to a 1581, or from a 1571 to a 1571, etc.). It takes everything from the source drive, including the format, and plops it on the destination drive. You need to use "Fcopy", and copy the disk file-by-file. The BAM is different between the drive types. The ONLY copy program that can copy from one drive type to another drive type is Maverick, because it has a way of making up the difference. Mcopy = Mirror copy.

Jeff Nit-picks: MCOPY will copy from Hard drive to RAMLink To FD drive to Commodore drive, as long as the source and destination are the same type, whether the directory is a real 1581/71/41 or an emulation. It will also copy a 1541 disk to a 1571 partition or disk with no problem, but not the other way around since a 1541 can't handle tracks 36-70.

COMMIE SEARCH

D G H A E M M R E T A V O N M H H G X W N I S
F K S N G I S E D O R C I M E V I T A E R C C
P Q E I Q K B G P S U P E R C P U K R I S E O
X S J T T Q X G C R E P M I E W R P O F O S T
C W E Q A Z X M F V E N Y J M Z T T B L G R T
I E L L M D D Z I F U T X O O O N S I Y J A R
J X W X R Z I R W O A Z T P C U T G N W A P E
I G B N O F D L R R Q Z N E Y Y S R H M W I S
F R Z Q F D X J A D D R S C L O O F A Z W O H
F S Y J R Q I O U V M P Z A H Z Q M R M Q F T
Y K K A Z Z N Q K D O H R L D T Q U B Z D O I
D K H F C O P Y Y C I H N H J W L A R D F D Y
O G H V G L X X K M P Z S O N V H P O E D P R
S Z H N J A S P R N X M B O C U J B N E D H I
C I Q W Y F R E N T N H P N F H W X R U Y I C
E T F G D I D B U S O G D E R O D O M M O C N
T F X F N I N O T S E L G G E T T O C S I I X
U O X C R I J I D S W B G E R O F F E J D W B
Y R E Y B B P P J F U R A T S D A O L A Q M S
D P K G S D F R E D N E F X K D B R N A J E L
X S O T N A V R E S E H T M U A M I G A E I W
S T V C C I X I Y O Z H F W T G L Q U N P C W
O K C Y M P J Z J A D V I O M H I P K Z U M Y

AMIGA

CALHOON
CE SPOCK PRINCE
CMD
COMMODORE
CREATIVE MICRO
DESIGNS
FCOPY
FENDER
FORMAT
GOSUB
GOTO
HARD DRIVE
JEFF
JIFFYDOS
JUDI
KNEES
LOADSTAR
MCOPY
SKYRIDER
SUPERCPU
NOVATERM
SERVANT
ROBIN HARBRON
SCOTT
EGGLESTON
SCOTT RESH
VALIDATE

Solution on page 6

THINGS THAT MAKE YOU GO "HMMM"

- ☺ Depression is merely anger without enthusiasm
- ☺ Drink 'til she's cute, but stop before the wedding
- ☺ Eagles may soar, but weasels don't get sucked into jet engines
- ☺ Early bird gets the worm, but the second mouse gets the cheese
- ☺ I'm happily married - but my wife isn't
- ☺ I'm not cheap - but I'm on special this week
- ☺ I almost had a psychic girlfriend, but she left me before we met
- ☺ I drive way too fast to worry about cholesterol
- ☺ I intend to live forever - so far, so good
- ☺ I love defenseless animals - especially in a good gravy
- ☺ If Barbie is so popular, why do you have to buy her friends?
- ☺ If you ain't makin' waves, you ain't kickin' hard enough
- ☺ Mental Backup in Progress - Do Not Disturb!
- ☺ Mind like a steel trap - rusty and illegal in 37 states
- ☺ Quantum Mechanics - the dreams stuff is made of
- ☺ Seen it all, done it all, can't remember most of it
- ☺ Shake well before and after use
- ☺ Support bacteria - they're the only culture some people have
- ☺ The light at the end of the tunnel is a muzzle flash
- ☺ The only substitute for good manners is fast reflexes
- ☺ When everything is coming your way, you're in the wrong lane
- ☺ One nation, under God, with Liberty, large fries and a Coke to go

Emailed by Jerry J. Gossett

LOADSTAR LETTER #48

J&F PUBLISHING • 606 COMMON STREET • SHREVEPORT LA 71101

- COMMODORE NEWS
- COMMODORE VIEWS

Bulk Rate
U.S Postage PAID
Shreveport LA
PERMIT #85